Question/Answer Transaction Protocol
Functional Model

First Committee Draft, revised
September 18, 2003

NISO Committee AZ:
Networked Reference Services

NISO Committee AZ is developing a Question/Answer Transaction Protocol (QATP) to support exchange between Digital Reference systems collaborating in the processing of a question. This is one of a set of committee documents related to the development of the protocol:

- Overview
- **Functional Model and Topology** *(this document)*
- Use Cases
- Protocol Specification
- Implementor Guidelines and Recommended Practices

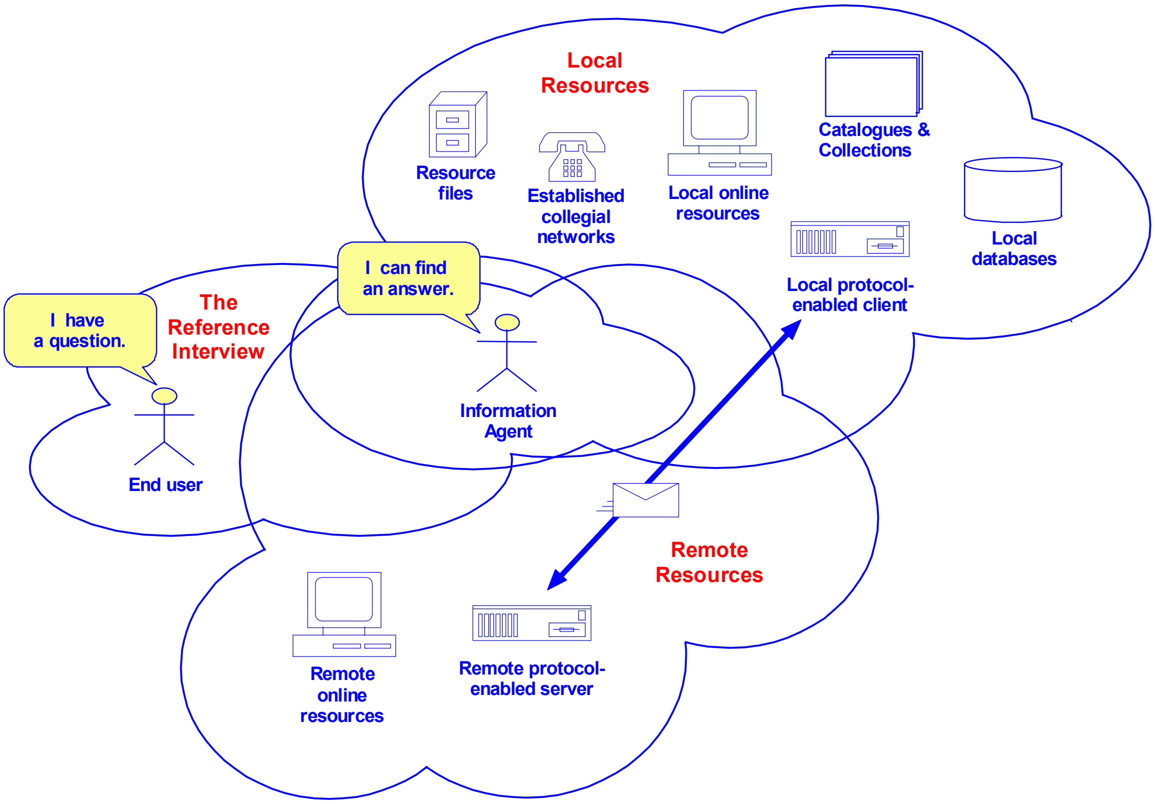
This document formally models the use cases (q.v.) that the protocol is required to support. Like use cases, functional models are typically developed prior to the definition of a protocol to help develop protocol requirements.

Contents

1. The reference environment

To satisfy users' information needs, information providers turn to an increasingly complex reference environment.

The traditional reference environment is an amorphous and highly complex environment consisting of a variety of resources: library collections, archival records, vertical and other resource files, catalogues, databases, and trusted collegial or expert sources. Given the ever-increasing selection of networked resources, the information provider now has available a multiplicity of online network options to add to the roster of traditional resources used to serve the information needs of end-users.



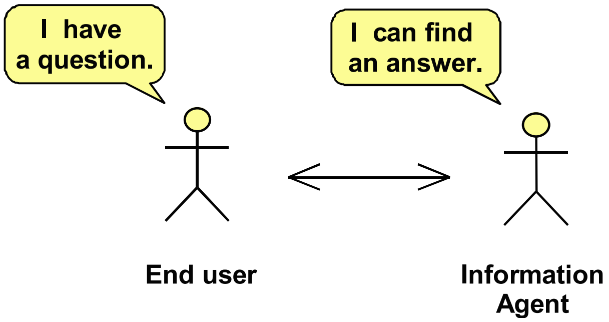
2. The functional model

Following the networked reference Use Cases, this functional model assumes that two systems ... **A** and **B** ... exchange QATP messages within discrete transactions following traditional client/server behaviour. Regardless of the number of systems involved in any complete reference interaction, QATP message exchange follows a bilateral model for each transaction. Thus, for example, in cases where three systems may be involved in answering a single reference question, message exchange will occur in separate transactions occurring between **A** and **B**, **B** and **C**, and **A** and **C.**

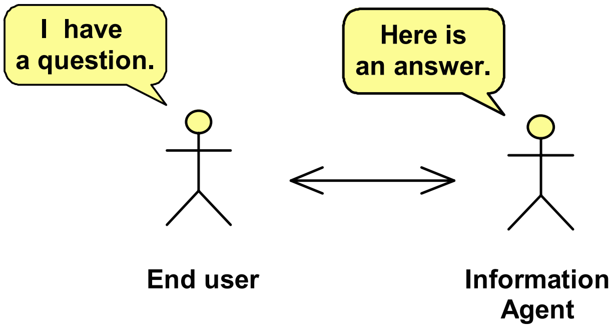
A more complete explanation of assumptions upon which reference transactions have been developed and modelled is presented in the Use Cases document.

2.1. The reference query

The primary model of a reference transaction is a simple reference question and answer exchange. The end user approaches an information provider and expresses an information need; and the information provider (a librarian, an archivist, an expert, a specialist, etc.) seeks to provide information to satisfy the user's need.

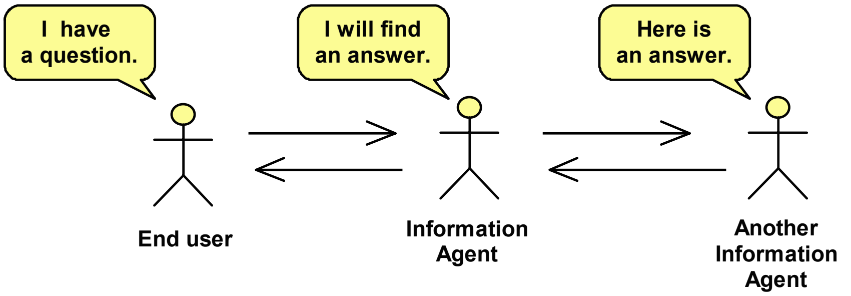


In some cases, the information provider may be able to answer the question from within the local reference environment.



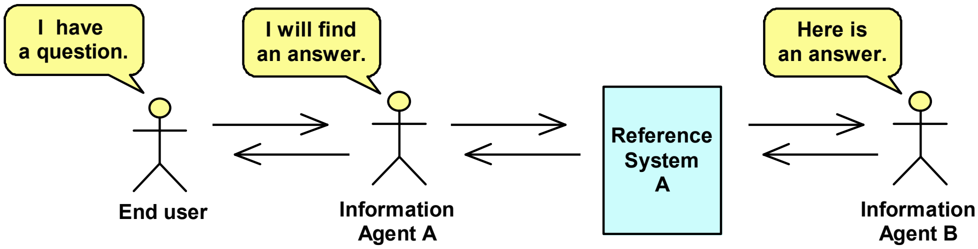
The end user may ask and receive the answer in "real time" (i.e., immediately or very soon after asking the question, such as when standing at the reference desk or when using a chat reference system); or the answer may be delivered "asynchronously" later via e-mail, telephone, etc.

In other cases, the information provider may send the question ... either synchronously or asynchronously … to another information provider to obtain an answer.

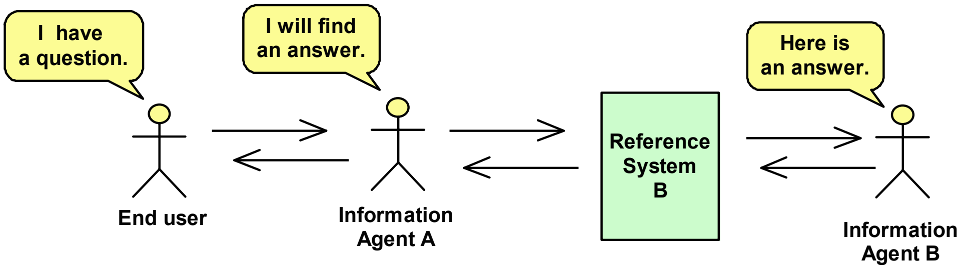


Although this type of case does not require use of a protocol, the transaction may also take place in an environment involving automated reference systems.

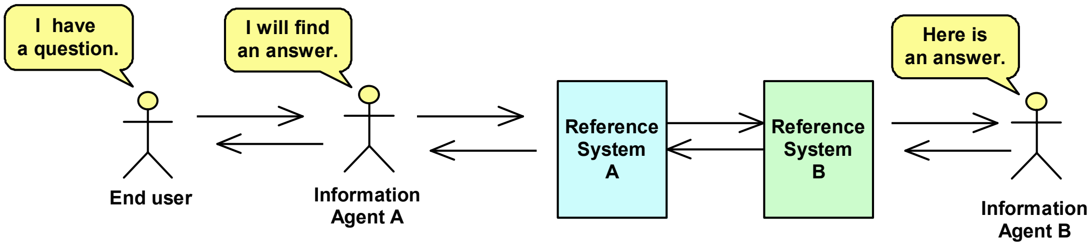
Information Agent A may have a Reference System A, which may act as a query management system to record, control, and send the question to Information Agent B.



Likewise, Information Agent B may have a Reference System B through which communication with A takes place.



If both A and B have Reference Systems, the reference transaction can flow through both systems.

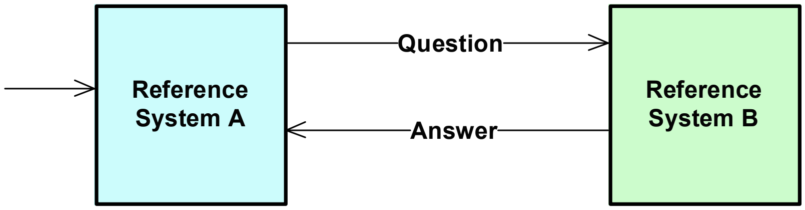


Protocol-enabled reference transactions are possible in such an automated reference environment.

Reference scenarios that may be handled by protocol messaging include simple and multipart questions and answers, clarification messaging, constraint negotiation, status reporting, referral, chaining, forwarding, etc. In fact, protocol messaging can be modelled to respond to or contain most reference scenarios. The illustrations that follow delineate these protocol-based scenarios by showing the reference transaction processes in the order in which they occur. In all cases below, a question from an end user prompts a protocol message exchange; the end user's question is denoted by the incoming arrow to the left of Reference System A.

2.2. Simple Question/Answer via Protocol

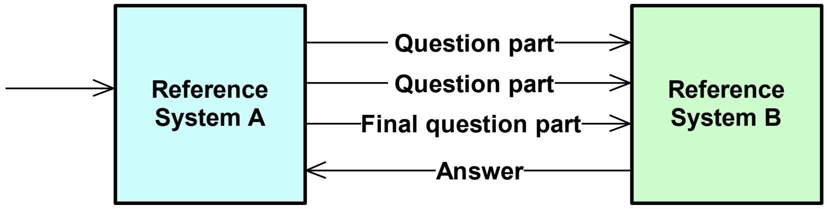
In the simplest of protocol-based system-to-system exchanges, a question is sent from A (the “client” system) to B (the “server” system); and an answer is returned from B to A.



In this model, the intellectual work of formulating the question and entering it into System A, and the intellectual work of answering the question and entering the answer into System B, reside outside the two systems. The systems themselves engage in an automated exchange governed by rules and procedures dictated by the protocol and implemented on both the “client” and “server” machines.

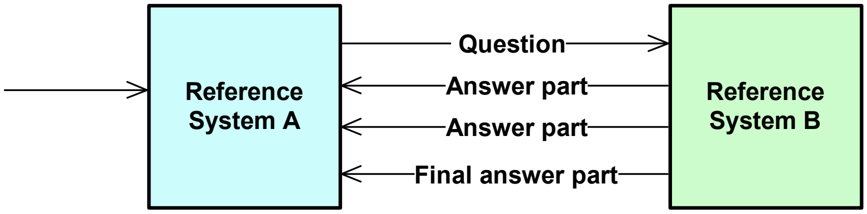
2.3. Multipart question

A question sent between two systems using the protocol may be sent in parts, rather than all at once, for any of several reasons: all supporting information may not available immediately, the question has not yet been completely formulated, something was omitted from the first part, etc.

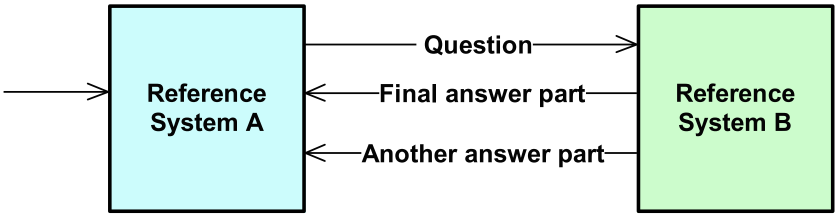


2.4. Multipart answer

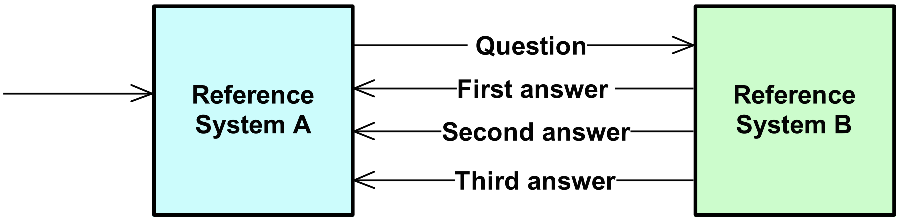
Answers may also be sent in parts.



B may provide what it thinks is the final answer part, and then discovers additional information and sends another answer part. A system may or may not designate a question or last question part, or answer or last answer part, as "final". Such designation in itself does not prevent additional question or answer parts from being sent later as part of the same transaction.

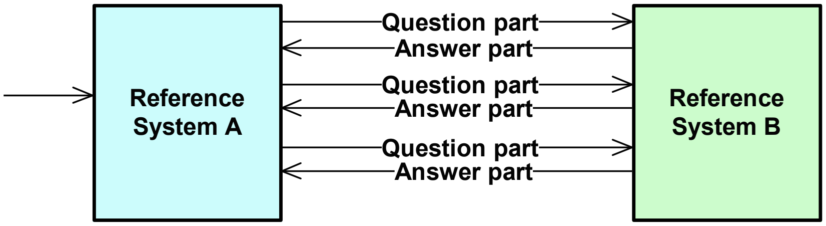


B may decide that the question is logically multiple questions and chooses to answer each identified question separately.

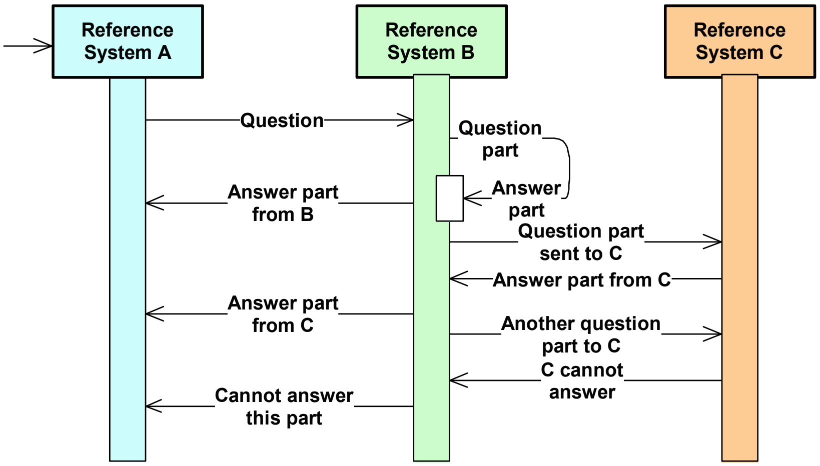


2.5. Additional multipart cases

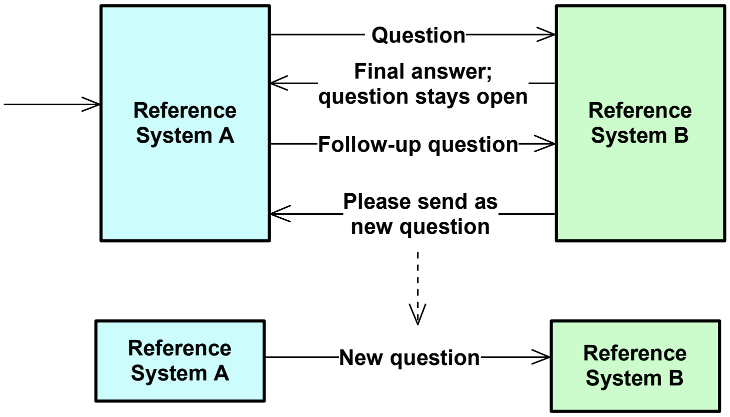
A multipart transaction may consist of several question parts with corresponding answer parts.



In the next example, B acts as an intermediary, answering some question parts itself, seeking answers to other parts of the question from third parties, and sending the answers back to A as they are received. In some cases, an answer may not be found for a question or part of a question; and a failure message must be sent. Here, C's failure to answer is reported through B to A.



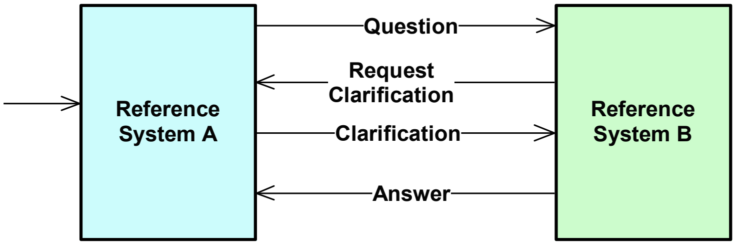
A transaction may contain what the recipient system B feels are two separate questions. B may ask A to begin a second transaction containing the second question.



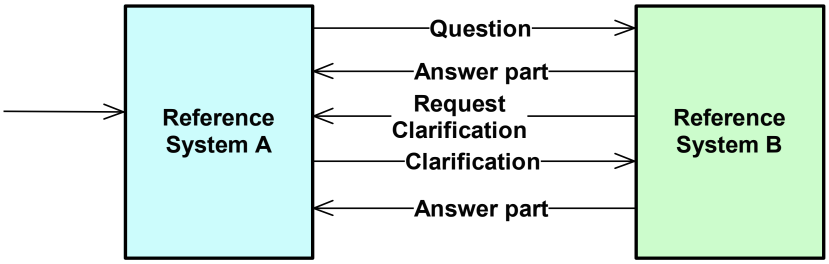
2.6. Clarification

In real-time reference (whether face-to-face or in a chat environment), the reference interview is an attempt to clarify the end user's question so that an appropriate answer may be obtained. In asynchronous reference, clarification may be sought through an exchange of e-mail or other type of messaging.

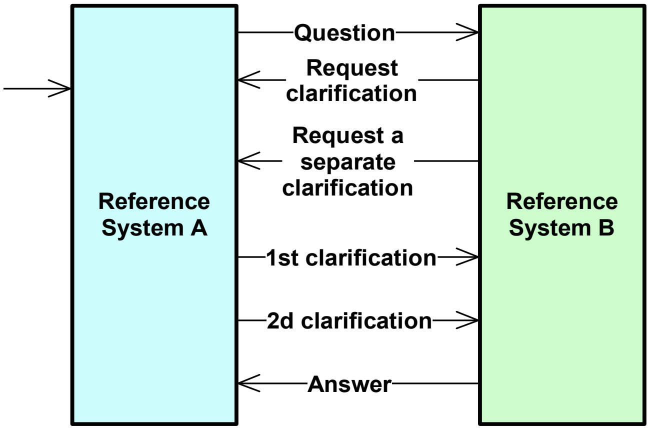
No matter the mode, clarification of questions is often needed. Simple clarification within a protocol transaction takes the following form:



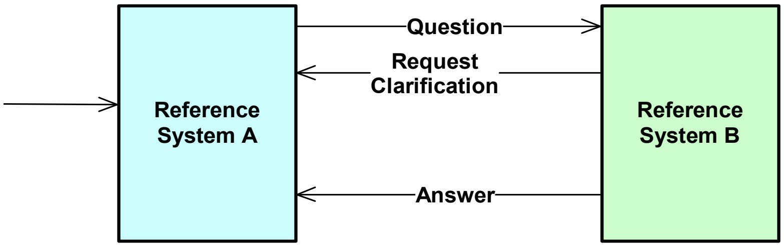
Clarification may occur before an answer is provided, or it may be needed after some answer parts have been sent.



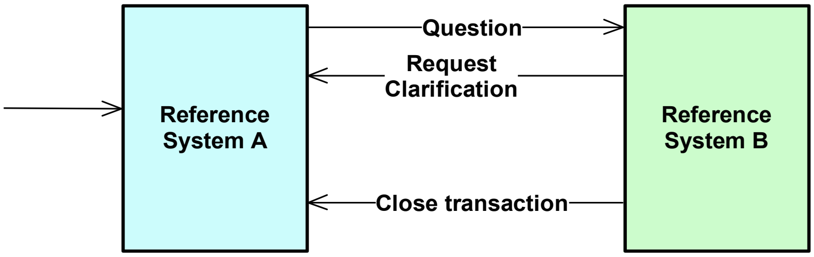
More than one clarification may be requested. Clarification requests and answering clarifications are associated using identifiers, so answering clarifications are not required to be sent in any sequence or order.



If a request for clarification goes unanswered, B may decide to proceed with the transaction and provide an answer anyway:



or B may close the transaction:



2.7. Transaction Progress

During a reference transaction, A may wish to query B on the status, state, or progress of the transaction; or either party may wish to change the status of the transaction so as to affect its progress.

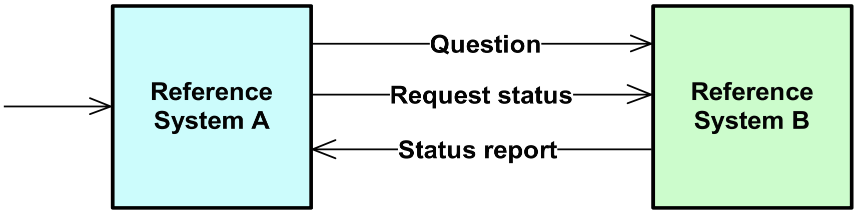
In the Use Cases document, conditions or states relating to the progress of transactions are defined:

> *Suspend* and *resume* refer to suspension and resumption of processing of the question, by the server.
> *Close* refers to ending the transaction (by either party).
> *Terminate* refers to premature closing of the transaction, by the server.
> *Cancellation* refers to termination, at the instigation of the client (i.e. the client request termination).
> *Failure* is the inability of the server to process a question.
> *Completion* refers to normal completion of processing the question, as well as of the transaction.

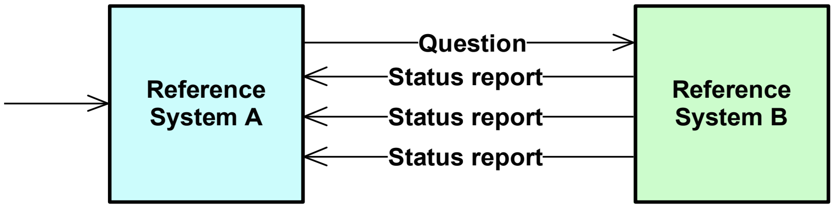
The first five conditions may happen at any time during the transaction. *Completion* occurs at the end of the transaction. Transaction progress messages require no acknowledgement from the recipient.

2.7.1. Status reporting

A request for status may be sent by either party.

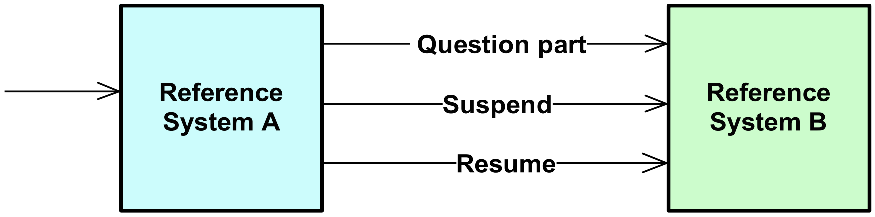


As well, one party may wish to send regular status reports to the other party during the time the transaction is active.



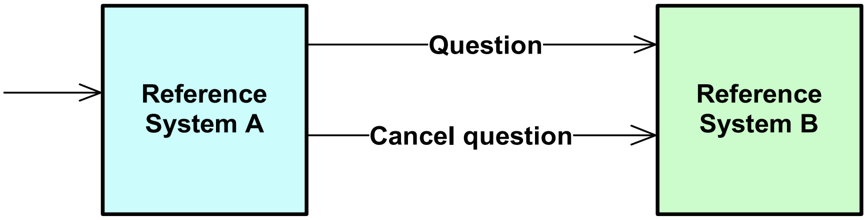
2.7.2. Suspend/resume

One party may need to suspend its involvement in a transaction for a period of time … perhaps due to other priorities, systems problems, etc.. A transaction that has been suspended is re-activated by a resume message.

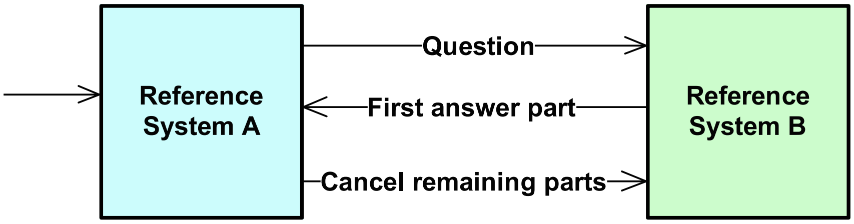


2.7.3. Cancellation

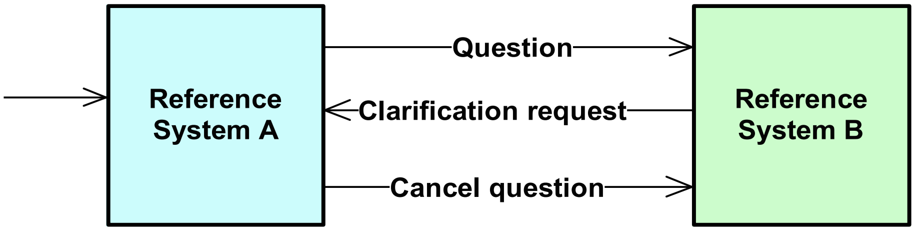
In a simple cancellation scenario, A may send a question and then (regardless of elapsed time) may decide to cancel the question before receiving anything from B,



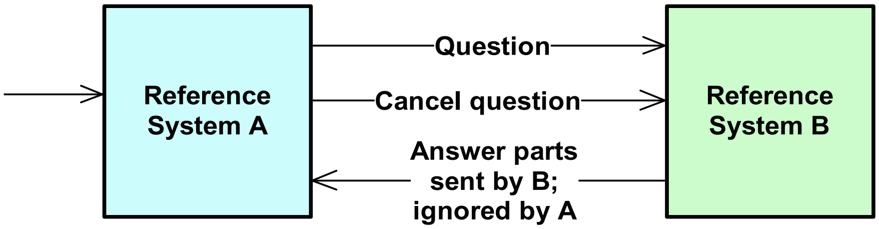
or A may decide to cancel after B has already sent an answer part but before B has completed the answer.



B may require a clarification; but, rather than supply a clarification, A decides to cancel.

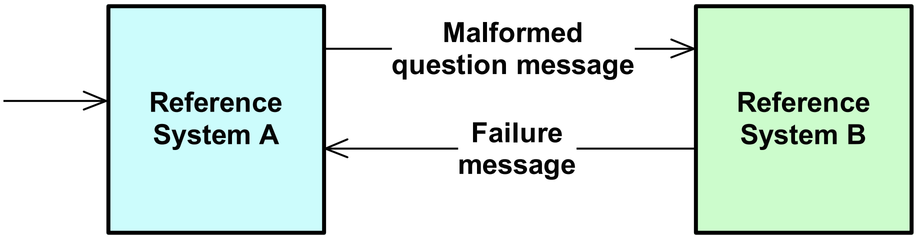


Cancellation messages change the status of the transaction, but do not stop it. Thus, messages may continue to be sent by B after A cancels the transaction. There is no requirement, however, for either party to acknowledge or otherwise take into account any messages sent after a cancellation message has been sent and received.



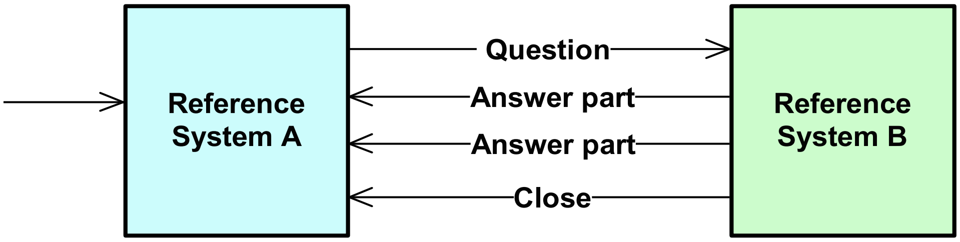
2.7.4. Failure

A failure occurs if the recipient server cannot process the message received. This may happen due to server failure of some sort, or it may be caused by a malformed message.



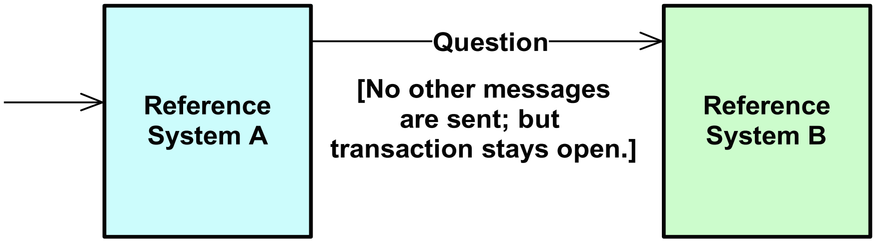
2.7.5. Completion/close

A true stop to a transaction is achieved by sending a close message. Messages sent by either party after a close message has been sent are considered to be out-of-band and may be ignored by the recipient.



2.7.6. Never-ending transaction

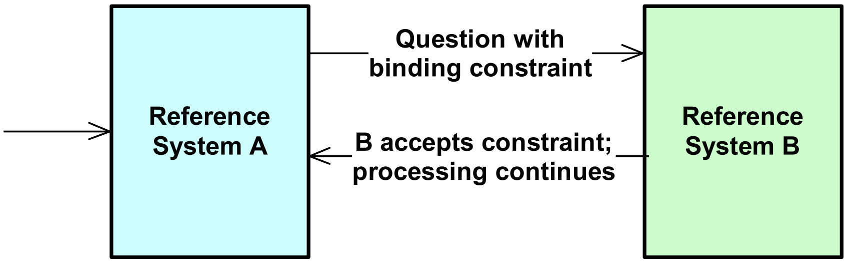
If neither party closes the transaction, then it remains open. This may be desirable in research scenarios, e.g., where complex multi-part queries with lengthy response times are being handled.



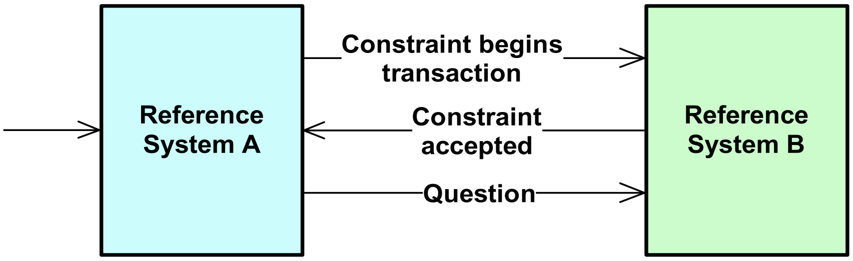
2.8. Constraint

Both questions and answers may have constraints placed upon them. There are two types of constraints: those which require an answer, agreement, or consent from the other system before a transaction may proceed (*binding* constraints); and those that do not require an answer (*informational* constraints).

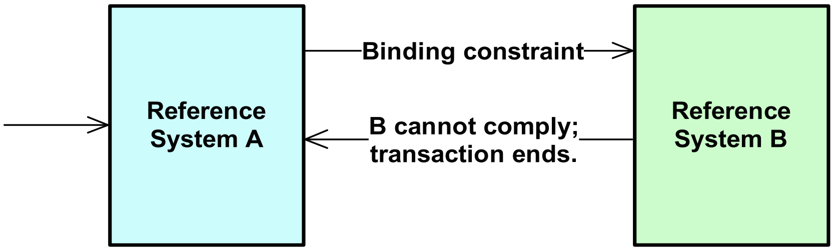
A question may have an accompanying binding constraint, e.g., needing an answer in a specified time. B may accept, in which case the transaction continues.


Reference System A
Question with binding constraint
B accepts constraint; processing continues
Reference System B

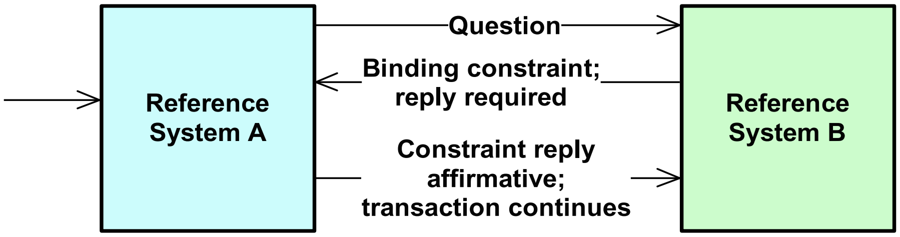
Alternately, a question may be preceded by a constraint, such as agreeing *before* the question is sent to not archive the question. B must accept the binding constraint for the transaction to proceed.



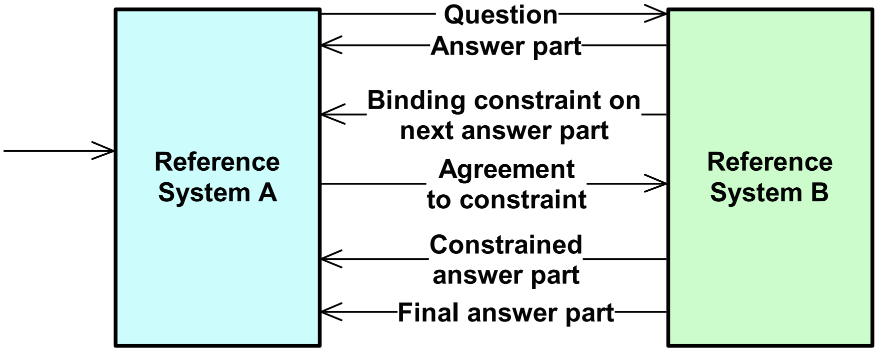
If B cannot comply, then the transaction will end.



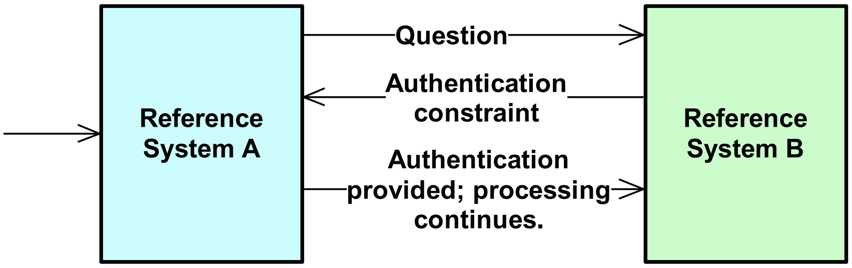
An answer may have an accompanying constraint, such as inclusion of a passage covered under copyright which the recipient must respect. Alternately, an answer may require that a constraint be agreed to *before* the answer is sent, such as agreeing to not archive the answer. If the constraint is not accepted by A, then the transaction may be closed by B.


Question
Binding constraint; reply required
Reference System A
Reference System B
Constraint reply affirmative; transaction continues

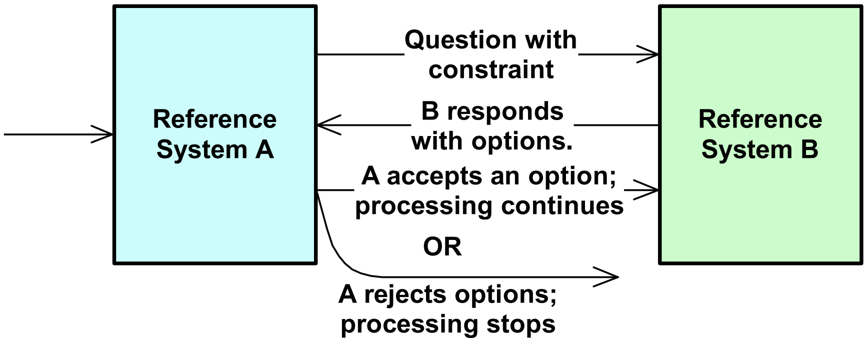
A constraint requirement may come midway in the life of a transaction.



Either party in a transaction may be required to authenticate themselves in response to a constraint before processing can continue.


Question
Authentication constraint
Reference System A
Reference System B
Authentication provided; processing continues.

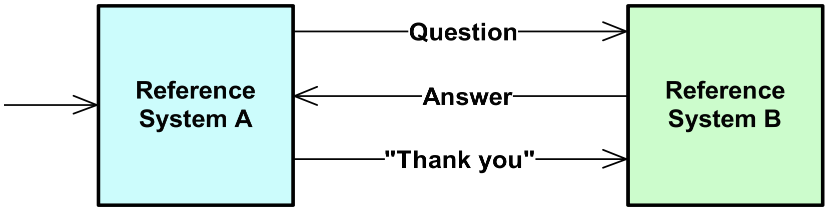
The introduction of constraints may lead to negotiation of various options before the conditions of a transaction are agreed to by both parties. An example would be negotiations regarding fees for services.



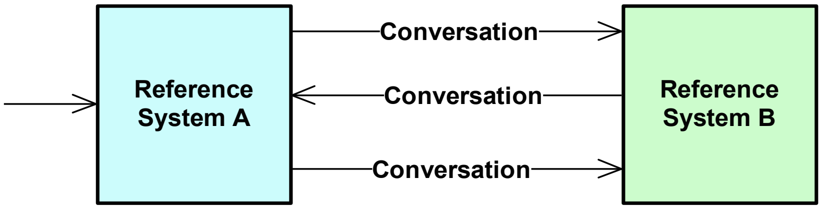
Many varieties of constraints are possible during the course of a reference transaction. They may be informational, speak to known conditions, be initiated by either party, accompany either questions or answers, apply to only parts of a transaction, or occur more than once during a transaction. They may involve formal authentication routines, or they may trigger negotiations of conditions. The Use Cases document presents a number of these constraint scenarios.

2.9. Conversation

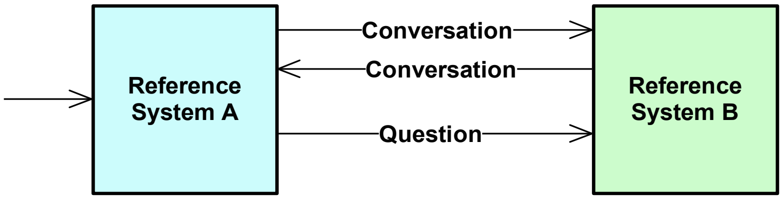
Conversation messages that add nothing of value to the transaction itself may occur. Such a message may be considered “in-band”, or part of the transaction (such as saying "Thank you" for an answer),



or “out-of-band” if they occur without any reference exchange being present.

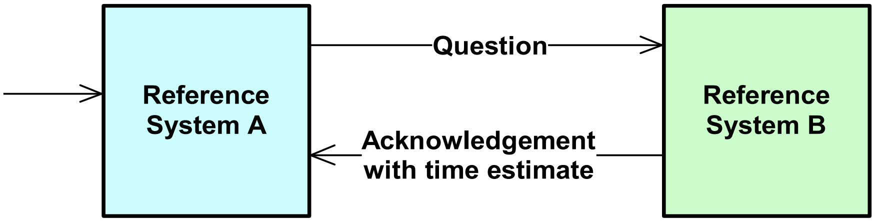


In real-time reference transactions, conversation messages may precede the statement of the question. Protocol-based transactions may behave in similar ways.

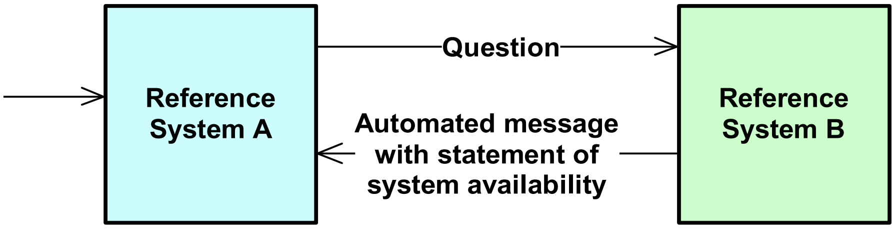


2.10. Question acknowledgement

Acknowledgements of questions received may be provided, either as a matter of course or to provide information such as a time estimate for completion of the transaction.

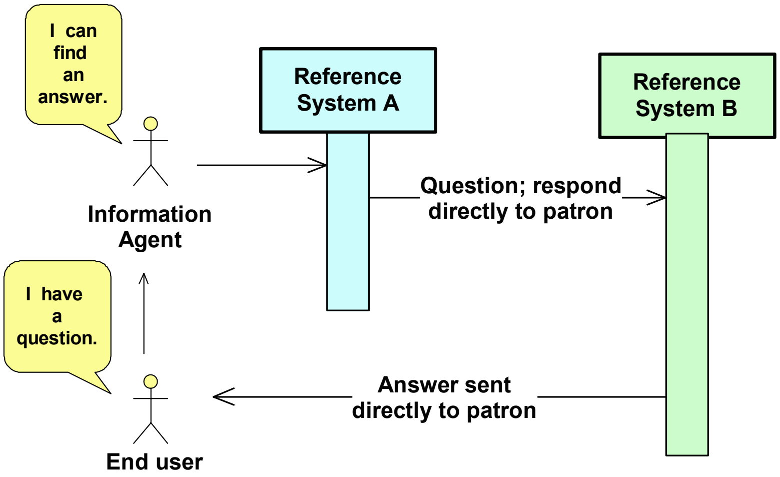


B may also send an acknowledgement in cases where system availability needs to be explained, e.g., to flag periods of planned server downtime.

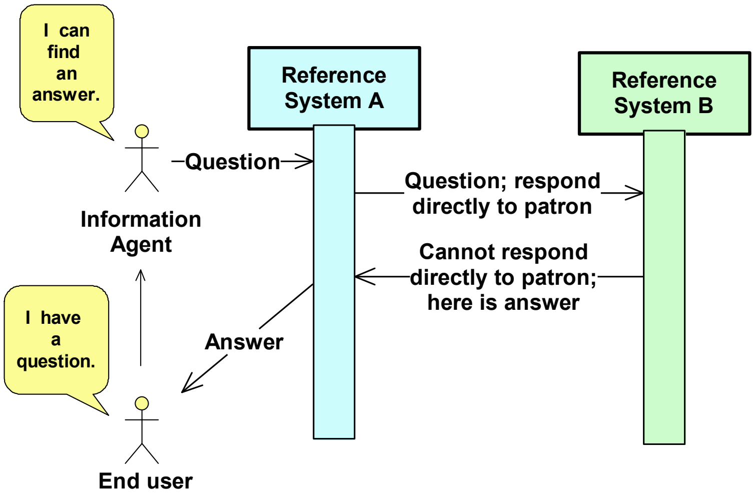


2.11. Reply to patrons

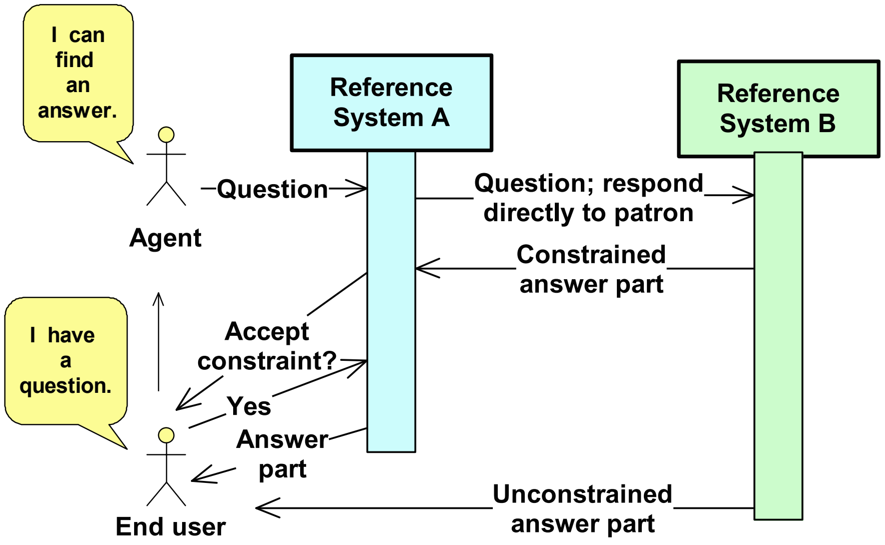
In reference transactions, it sometimes happens that information provider A would like a response to go directly to the patron from the answering system.



Sometimes the answering system can support this and sometimes, for a variety of reasons (technical, legal, etc.) it cannot.

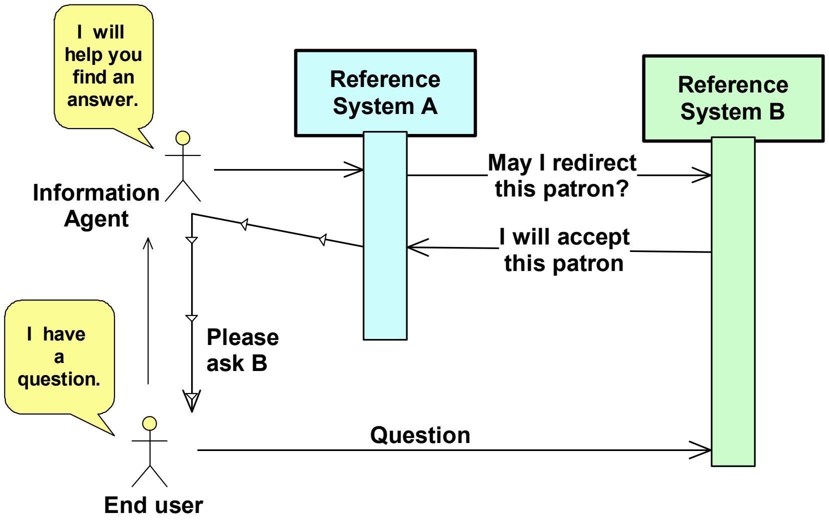


Sometimes part of the answer may be provided directly to the patron, and part cannot. This will happen if part of the answer has one or more accompanying binding constraints.



2.12. Patron redirect

A patron redirect occurs when A, through protocol messaging, formally asks B to accept responsibility for the end user and their question; and B agrees. A then instructs the end user how to contact B, and the end user begins a new reference session with B.

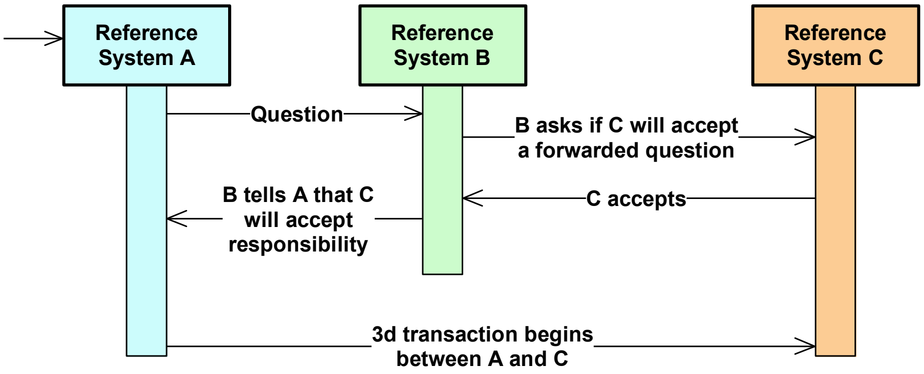


Of course, the patron may end up not contacting B at all; or B may decide to hand the patron on to reference system C. B may also, either as standard procedure or to satisfy a specific request from A, copy the resulting answer to A after the session with the patron is completed.

2.13. Forwarding

Forwarding occurs when B, rather than handling the question in any way, wishes to pass responsibility for the question to another party. Forwarding is achieved when a third party C agrees to become responsible for the question *before* B responds to A.

In the forwarding scenario, the first transaction between A and B is followed by a second transaction between B and C. The second transaction contains the question from the first, along with a request for C to accept responsibility. If C accepts, then the transaction between B and C is closed. B replies to A, stating that C will handle the question; this closes the transaction between A and B. A then initiates a third transaction with C, referencing the prior communication between B and C.

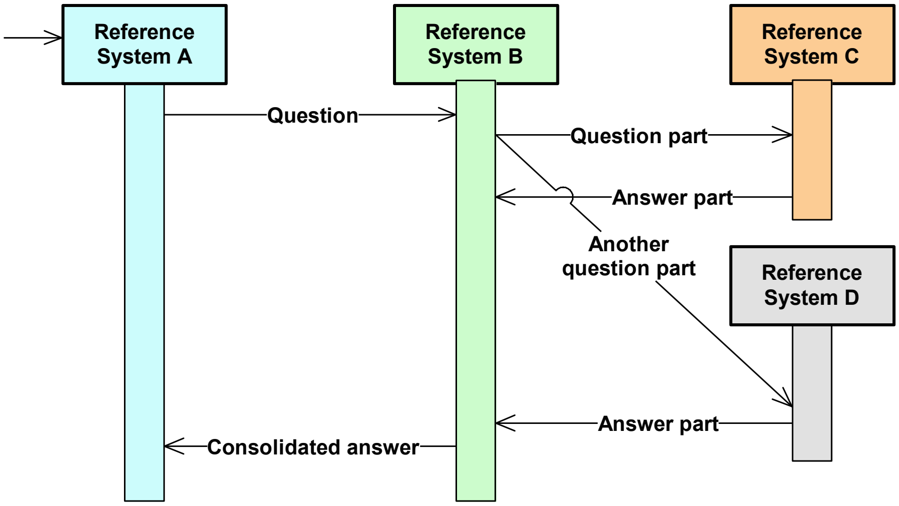

Reference System A
Reference System B
Reference System C
Question
B asks if C will accept a forwarded question
B tells A that C will accept responsibility
C accepts
3d transaction begins between A and C

Variations are possible:

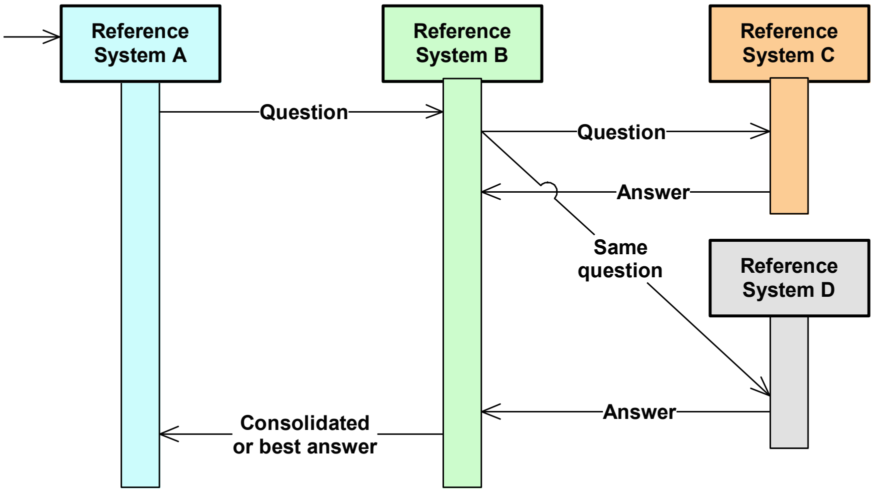
- A may never contact C.
- If B receives subsequent parts of a question after sending a forward request to C, B may choose to keep or discard, and forward or not forward, the additional parts.
- C may reject a request to forward, in which case B must either find another party to which to forward the question, or must tell A that it cannot answer the question.
- Upon receipt of the request to forward, C may decide to send an answer back to A through B, rather than accepting the forward and then dealing directly with A..
- B must advise C of any applicable constraints so C may make an informed decision to accept or reject the request.

2.14. Multiple questions

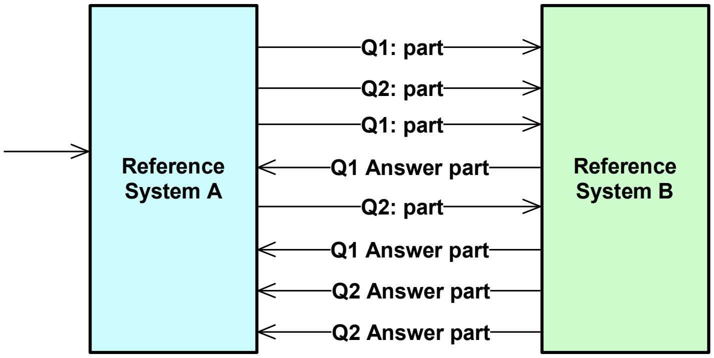
A variant on multipart questions and answers occurs when system B decides to not answer the question itself and enlists assistance of system C and system D in answering. B sends question parts to C and D, which respond; and B subsequently consolidates and sends the answer back to A.



B may choose to multicast the same question to multiple servers, receive answers back, and then consolidate or otherwise choose the best answer to send to A.



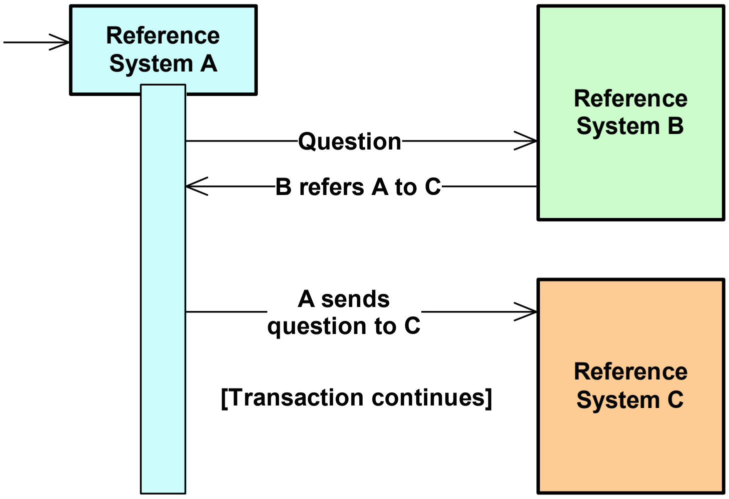
Multiple asynchronous questions may be sent from A to B. Each question begins a separate transaction, the messages of which then proceed in an intermingled fashion. In the following, Q1 and Q2 are two separate questions transmitted in the sequence shown. The question and answer parts in such scenarios carry identifiers so they can be appropriately associated.



2.15. Topological scenarios

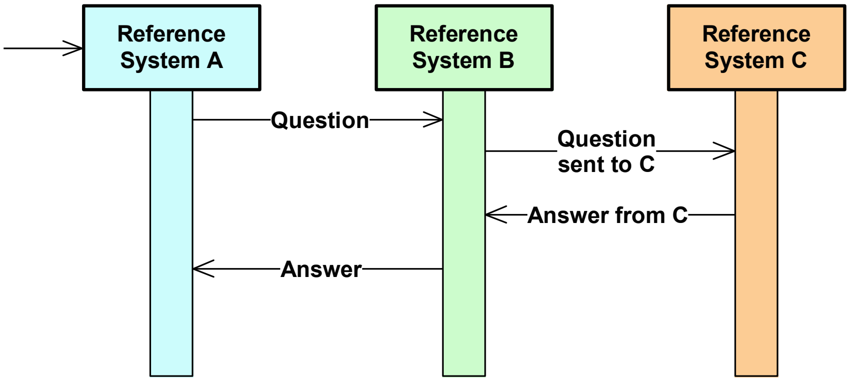
2.15.1. Referral

Referral is frequently performed in reference work. It follows the same sequence in protocol-based reference as it does in face-to-face reference.



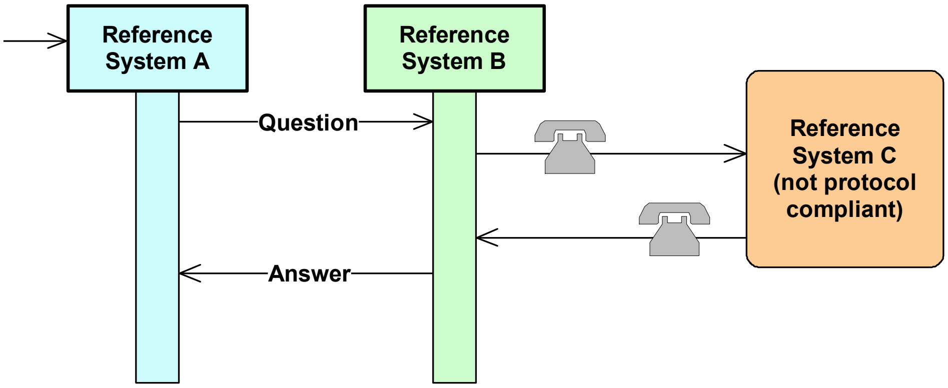
2.15.2. Chaining

In chaining, A sends a question to B. B, acting as intermediary, sends the question to C for response and sends the resulting answer from C back to A.



2.15.3. Gateway

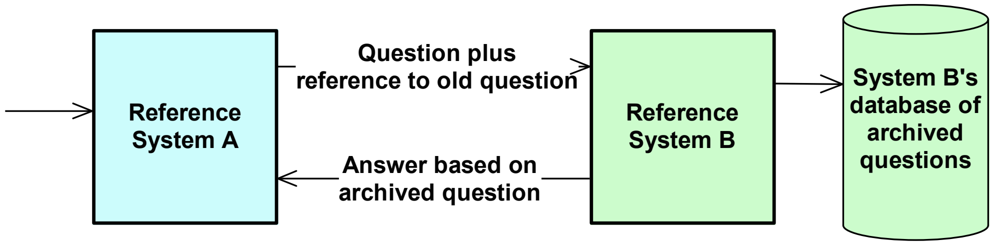
A gateway scenario occurs when a third party involved in a reference exchange is not protocol compliant. In this instance, a protocol-compliant system acts as an intermediary between the client system and the non-protocol-compliant system.



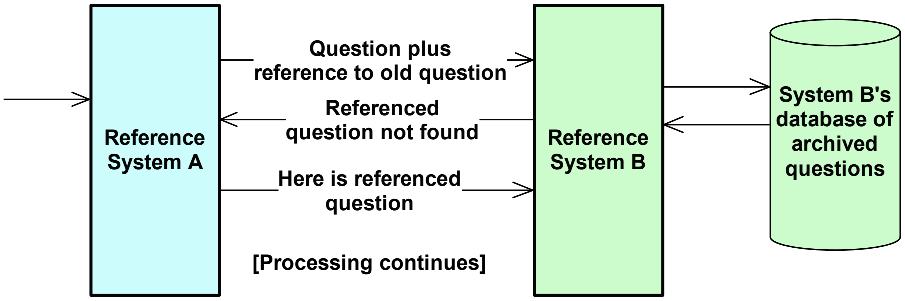
2.16. Referencing archived transactions

Reference systems may keep archives of transactions following local policies, practices, and laws. Whether a transaction can be retrieved from a local archive will not necessarily be known by another information provider referencing an earlier transaction in a current transaction.

Sometimes an earlier transaction will be retrievable.


Reference System A
Question plus reference to old question
Answer based on archived question
Reference System B
System B's database of archived questions

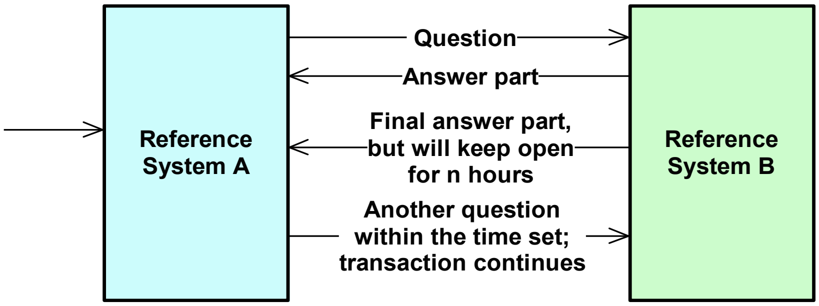
Sometimes an earlier transaction will not be retrievable. In this case, A may close the transaction, or may tell B to process anyway, or may provide B with the earlier transaction so that processing may continue.


Reference System A
Question plus reference to old question
Referenced question not found
Here is referenced question
[Processing continues]
Reference System B
System B's database of archived questions

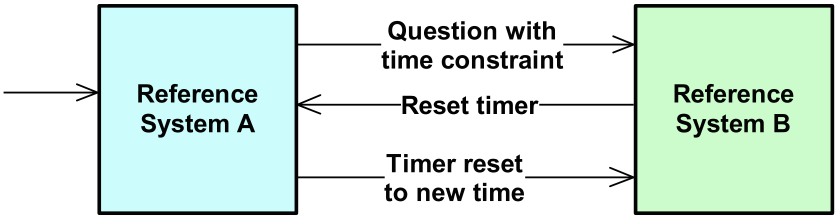
2.17. Timers

On occasion a timer may be set by either the client or the server system. A timer may indicate a deadline by which time an answer is needed, or a timer may flag that a response to a protocol message should be made within a certain timeframe.

In the following scenario, B sends an answer part saying "this is the final answer, but we'll keep the transaction open for an additional hour; if we don't hear from you, the transaction will then be closed." **A** sends another question-part within the hour and the transaction continues.


Reference System A
Question
Answer part
Final answer part, but will keep open for n hours
Another question within the time set; transaction continues
Reference System B

A system can request that a timer be reset if it wants to keep a transaction open, or if it needs added time to respond.



In some cases, when a timer runs out, the transaction will remain open; in many cases, however, it will close.